Animated Software Training Via the Internet: Lessons Learned

Carol J. Scott,
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive M/S 301-270
Pasadena, California, USA 91109-8099
(818) 393-0551
carol.j.scott@jpl.nasa.gov

Abstract--The Mission Execution and Automation Section, Information Technologies and Software Systems Division at the Jet Propulsion Laboratory, recently delivered an animated software training module for the TMOD UPLINK ConsolidationTask for operator training at the Deep Space Network. The animated training software is presented as a Command Subsystem simulation for on-line instruction. The animation was developed as a prototype (a paper [1] describing the initial development of this animation was presented at the 2000 Aerospace Conference.) From development through delivery, significant design changes were made to keep pace with the changing Command formats and techniques being developed for the Deep Space Network, thus enabling training to remain concurrent with development.

TABLE OF CONTENTS

1. INTRODUCTION

The original objective was to construct a prototype for web-based delivery of mission software training that provided a mechanism for operators and engineers to master unique, operations-specific software subsystems. This training procedure had to accommodate operator and operations schedules, without interfering with hardware or software delivery schedules. On-line delivery of the training software made the tool available on demand for initial training, operator practice, and peer review of operating procedures.

Initial exposure for the user population occurred during the development of the animated training package with feedback used both for training modification and for system enhancements. The training prototype evolved into a software simulation utilizing the capabilities of the vector animation development software (Flash). An animated simulation remained on-line, in a user-ready format, during the development cycle. Frequent access (measured hits on the site) indicated significant early interest in the tool, and provided an avenue for user feedback to incorporate enhancements into the design. The formal introduction was in the form of a demonstration to the target operations audience; it presented a mature development that had evolved dynamically thanks to inputs from developers and system engineers.

2. CHANGES IN LOOK AND FEEL

The look and feel of the delivered training animation differed significantly from the original design due to continuing enhancements for optimizing on-line delivery. Figure 1 depicts the original design. Drawbacks to this design included color scheme, clumsy buttons, limited options, and lack of instructional information. Too many clicks and page changes were required to get smooth information flow. There was no convenient place to place shortcuts that could be maintained in the same location on all scenes of the animation. Color became a problem when system developers began providing screen dumps that differed in appearance from their original images. They used personalized palates during their login sessions, which produced graphical user interfaces (GUIs) that differed from their original color schemes. By using a multi-colored format, the animation could support most any color GUI.

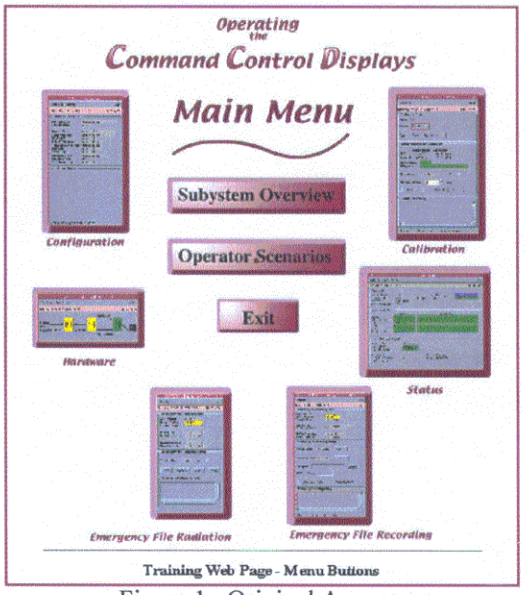


Figure 1: Original Appearance

Figure 2: Delivered Appearance

Figure 2 depicts the delivered version of the animation. This version is more flexible, providing information without screen changes. The words "Start Here" direct the user where to start. By rolling over the first green button, a pop-up appears describing the Main Menu (Figure 3.), which includes a small image of each of the Button Bar buttons and information on how they are used.

Figure 3: Roll-over Pop-up

By clicking on the top yellow button, "Show Display Summaries" on the Button Bar, an instruction appears for getting information on each of the GUI buttons. When the instruction is present, simply roll over a GUI button for an instant summary of the display's capabilities and usage (see Figure 4.)

Figure 4: Display Summary Pop-up

When you are ready to begin training, click the second green button (seen in Figure 2) to see your startup environment (Figure 5.)

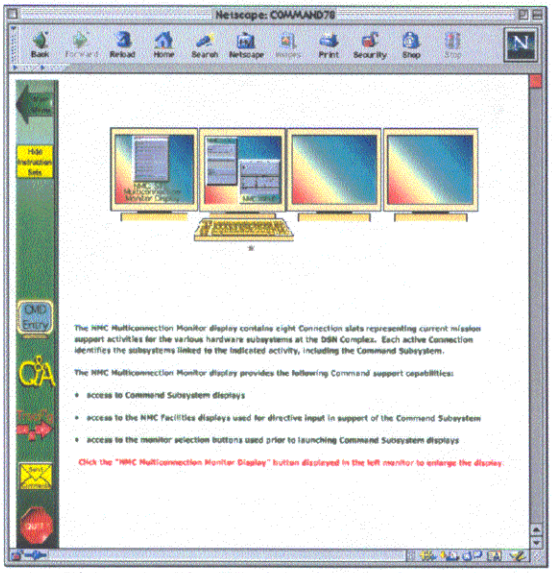


Figure 5: The Workstation Environment

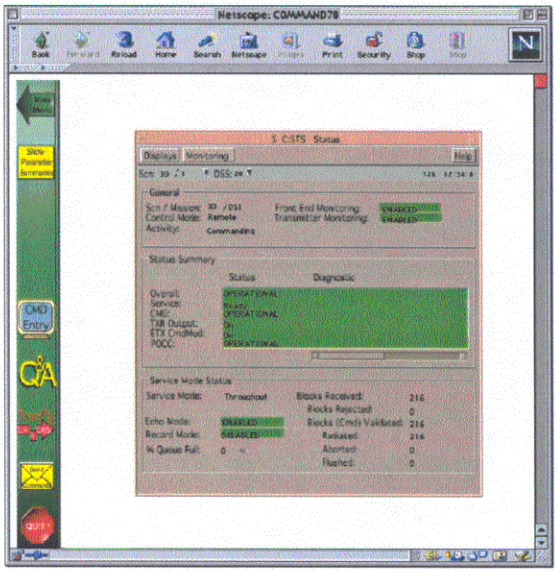


Figure 6: Status Display

By clicking the control display, in the left monitor (see Figure 5,) a full-sized GUI appears with instructions for bringing up the Command Subsystem menu. Most users begin with the Status display (Figure 6) for monitoring commanding activities during the track; it is left to the learner to decide the sequence of the training.

The original animation had a specific sequence that could be changed with a cost of loosing scenario context. For advanced technical software training of this type, it is assumed that the student is a sophisticated user, and familiar with station hardware and interfaces. The information presented must be clear and comprehensive, subtle in nature allowing the basics to be available when needed and not obstructing the more advanced nature of system operation. By enabling the student to select the course of action, the student could maneuver immediately to the point of interest without having to suffer through layers of already familiar material.

Changes to individual GUI displays were incorporated to give them a realistic feel. Initial attempts at building menus treated individual options as separate clunky buttons overlaying the rectangular menu background. Exploring more precise duplication methods resulted in near facsimiles of subsystem menus and buttons.

Buttons on the displays produce realistic results. When desiring to simulate disabling of the Transmitter monitoring, one can do it on the simulation just as it is performed on the subsystem (see Figure 7.) In addition, values update to simulate the Project Operations Control Center (POCC) is radiating commands.

Figure 7: Working Menu Option

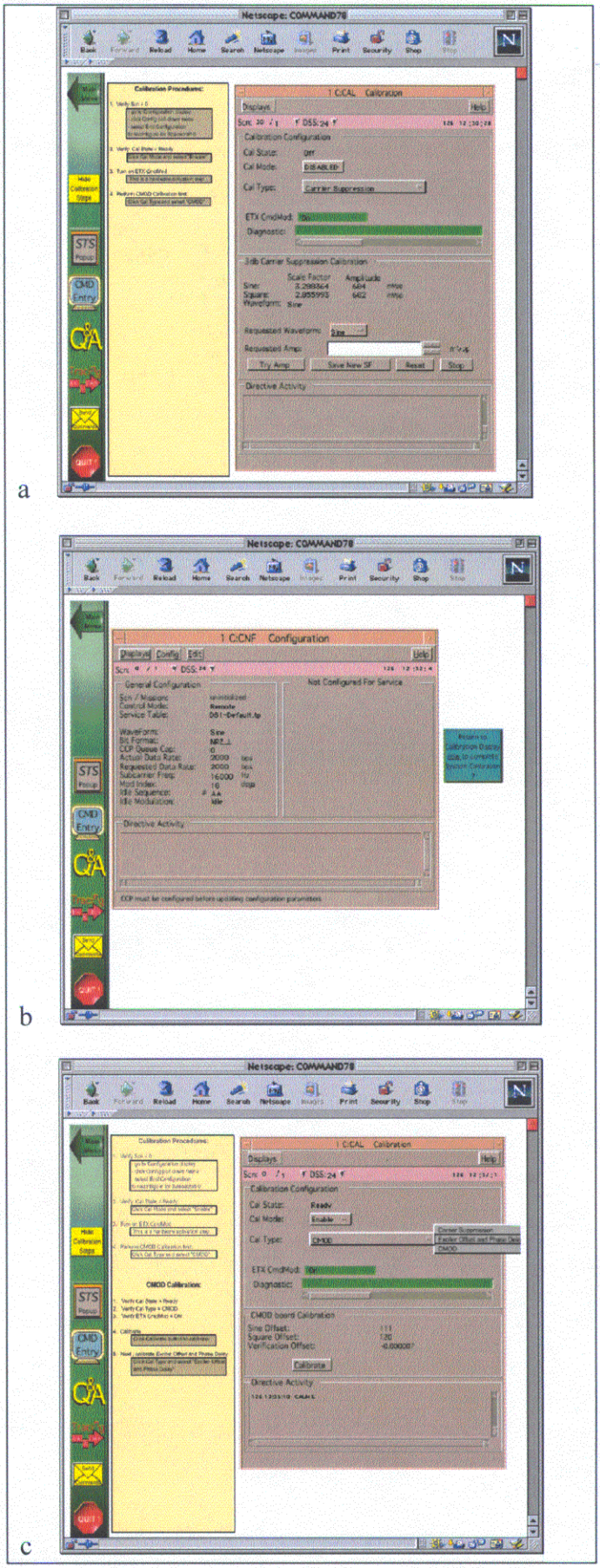
3. CHANGES IN CODE STRUCTURE

Objects that performed action (movement or created results) were better managed as movie clips. Movie clips appear invisibly for call or recall to immediate action. A justification for using movie clips is that the main timeline is less populated, making it easier to manipulate. The use of movie clips limits the timeline to fewer frames. Each movie clip contains its own unique timeline. Our original calibration time line was several hundred frames and nearly totally sequential and inflexible. When converted, the main timeline called three movie clips, and the movie clips, in turn, called several additional movie clips. Consequently, the Calibration display's time line was 80 frames, and many of those frames were blanks employed for readability.

Layering was an exercise in patience. It was easy to add objects to the time line, however, if any object needed attention (editing, moving, locating, etc.) it was not easy to isolate and manipulate unless it was on a layer of its own. Therefore, each object (like buttons, images, or variables) or object group (like masks, static text, or navigational instructions) was given its own layer. This made development time-consuming, while allowing debugging and maintenance to become nearly effortless. The Calibration display's timeline contained 102 layers (see working display scenario in Figure 8.)

Figure 8 describes: a) the initial calibration display with instructions to, b) set a spacecraft value of 0, c) configure an internal board, d) configure the processing hardware, and e, f, g) perform signal calibration. A student apprentice (APT) created the spectrum analyzer cartoon.

For non-programming engineers and trainers, Flash is an excellent choice for creating simulations because animation evolves in such a manner that action choices are presented to the developer as options to be assigned to objects or frames. By carefully sequencing objects in frames, selecting strategic actions, and learning to use other inherent capabilities of the program, a "thousand words" can be easily reduced to an animated picture.



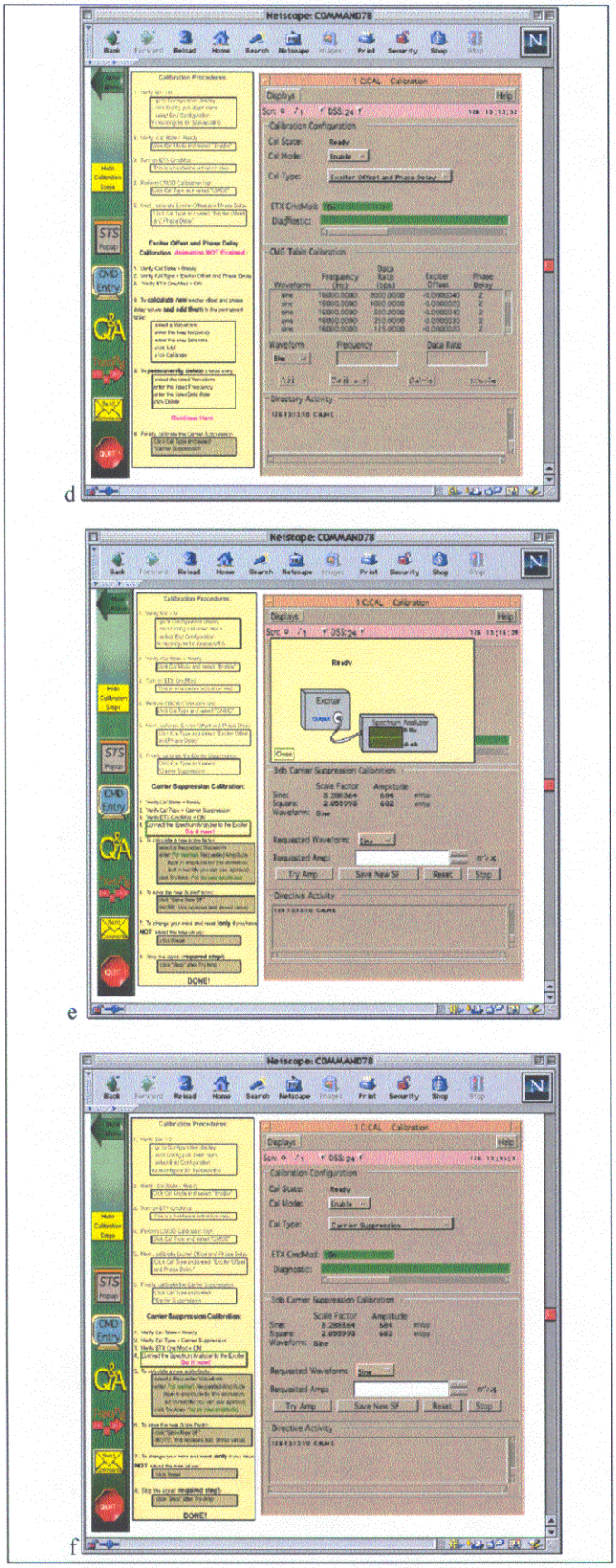


Figure 8: Functionality in Layers and Frames

4. CHANGES IN TRAINING SCENARIOS

On-site subsystem training was planned as a two-phase effort:

1) Initial classroom training - preparation for the deliverable Command Subsystem.

2) Hands-on instruction using the operational Command Subsystem.

The animated Command simulation was developed as a learning tool to provide access to realistic displays providing familiarity with operation and to gain proficiency in using them.

When subsystem delivery dates were in close proximity, all resources were dedicated to test and integration at the three DSN sites. Station personnel required immediate training, and remote training via video conferencing was considered. A training process, to be successful, requires that the trainee gets to "do it", rather than watch it. Previous remote hands-on software training experiences had been partially successful, although painfully time-consuming with screen visibility fuzzy and audience actions not being visible by the instructor.

A decision to use Microsoft's NetMeeting with a separate voice line for communication and verbal instructions was made. A NetMeeting session was configured to enable an Internet browser to run the animated simulation on a host machine at JPL, while three machines at Canberra, Australia, each logged into the JPL host machine. Fourteen DSN operators participated at the meeting huddled around the three machines. The participants at Canberra took physical control of the program running on JPL's host

machine. A separate voice line used speakerphones to avoid impairment of response time and to enable everyone in attendance to hear and speak. The instructor gave voice instructions and monitored the action as the operators took control of the simulation from their machines via their own mouse and keyboard. The slight screen delay was nearly imperceptible as the training proceeded. The entire session took a little over an hour and the response was extremely positive.

The response was so enthusiastic that all initial training at Canberra and Madrid was performed using the animated simulation via NetMeeting. Additional training on the real subsystem was provided during or prior to user acceptance testing.

4. Delivery Bonus

The benefits of a training simulation were apparent prior to subsystem delivery at the DST's Gladstone Deep Space Communication Complex. With the original delivery timeline, training would coincide with the end of integration testing. The subsystem was performing as expected and it was assumed that training would use the unused portions of scheduled subsystem test time. During the transition from test hardware to operational hardware, the usual unexpected problems were encountered. When the Training team arrived at the Goldstone complex, the workstation containing the new Command delivery was unavailable to trainers, as massive re-testing was the priority of the day.

Subsystem delivery required trained operators. On-site testing required constant usage of the subsystem, providing limited or no opportunity for operators to experience the actual subsystem procedural indoctrination and training. The training simulation, which was developed as a learning and practice module, was delivered via the Internet and in stand-alone mode via the browser (without Internet connection), and was utilized as the primary learning resource for trainers.

5. Conclusions

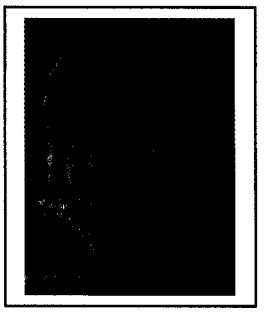
Lessons learned from the training development effort incorporate the resolution of problems in module design and structure, simplification of methods, version control, customer input, object control and orientation, planning, scheduling... the same design and development issues of a typical software development effort, with an entirely different focus. The Training focus is to make the operators' job easier, or at least, easier to learn. With that in mind, and by developing training concurrent with system development, there was a cross-pollination of developer-trainer-operator feedback. That customer focus is the key element to on-line training development.

Reference

[1] Carol J. Scott, "Vector Animation: Web-Based Software Training On Demand," 2000 IEEE Aerospace Conference Proceedings, March 18-25, 2000.

Acknowledgement

The Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration is carrying out the research described in this paper.



***Carol Scott**-- is a Lead Information Technology Specialist at the Jet Propulsion Laboratory where she has designed and developed software training for mission flight teams and DSN operators for the past ten years. She has a BS in Computer Science from Chapman College and a BA in English from the University of Nebraska at Omaha. She authored four papers on training development.*